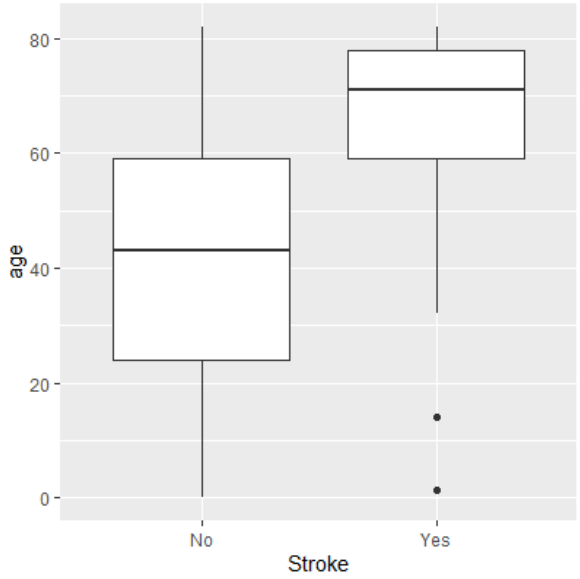
DSC 441 Assignment 5
Dhruv Borad (2049882)
Dataset link: https://www.kaggle.com/fedesoriano/stroke-prediction-dataset

A) Data Exploration

```
setwd("C:\\Users\\dhruv\\Documents\\csv files")
df <- read.csv("C:\\Users\\dhruv\\Documents\\csv files\\healthcare-dataset-stroke-data.csv")%>%
  rename_all(tolower)

df$stroke<- factor(df$stroke, levels = c(0,1), labels = c("No", "Yes"))
df$gender<-as.factor(df$gender)
df$hypertension<- factor(df$hypertension, levels = c(0,1), labels = c("No", "Yes"))
df$heart_disease<- factor(df$heart_disease, levels = c(0,1), labels = c("No", "Yes"))
df$ever_married<-as.factor(df$ever_married)
df$work_type<-as.factor(df$work_type)
df$residence_type<-as.factor(df$residence_type)
df$smoking_status<-as.factor(df$smoking_status)
df$bmi<-as.numeric(df$bmi)
```

```
> summary(df)
       id          gender          age        hypertension heart_disease ever_married         work_type
 Min.   :   77   Female:2897   Min.   : 0.08   No :4458     No :4666      No :1705     children     : 671
 1st Qu.:18605   Male  :2011   1st Qu.:25.00   Yes: 451     Yes: 243      Yes:3204     Govt_job     : 630
 Median :37608   Other :   1   Median :44.00                                           Never_worked :  22
 Mean   :37064                 Mean   :42.87                                           Private      :2811
 3rd Qu.:55220                 3rd Qu.:60.00                                           Self-employed: 775
 Max.   :72940                 Max.   :82.00
 residence_type avg_glucose_level      bmi                smoking_status stroke
 Rural:2419     Min.   : 55.12    Min.   :10.30   formerly smoked: 837   No :4700
 Urban:2490     1st Qu.: 77.07    1st Qu.:23.50   never smoked   :1852   Yes: 209
                Median : 91.68    Median :28.10   smokes         : 737
                Mean   :105.31    Mean   :28.89   Unknown        :1483
                3rd Qu.:113.57    3rd Qu.:33.10
                Max.   :271.74    Max.   :97.60
```

We can see presence of outliers with boxplots too



There is 201 *NA* values

```
> sum(is.na(df))
[1] 201
```

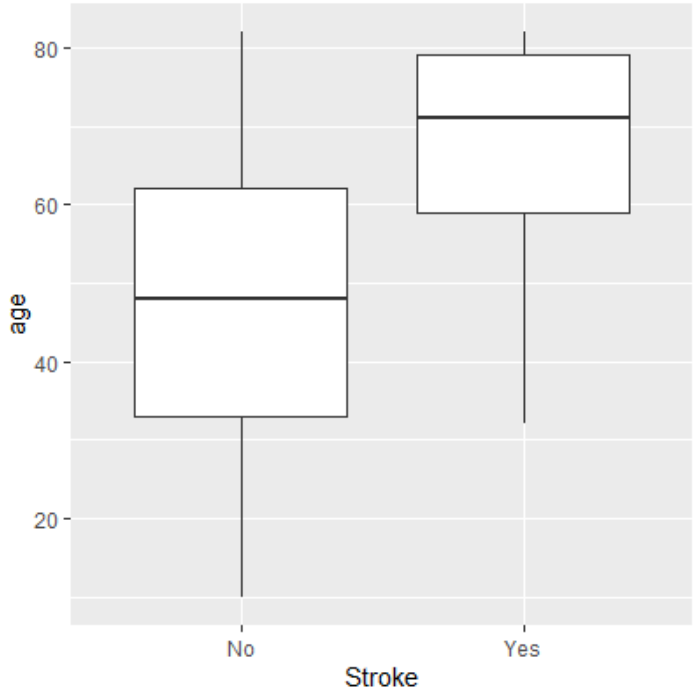
Removed all rows with *NA* values.

```
> df <- na.omit(df)
> # data cleaning
> sum(is.na(df))
[1] 0
```

Unknown in smoking_status means that the information is unavailable for this patient. Therefore, we are removing all Unknown labels.

df_num <- df
df_num <- df_num[!(df_num$smoking_status == "Unknown"),]

```
> count(df_num[df_num$smoking_status == "Unknown",])
  n
1 0
```



After removing the *NA* and Unknown labels we can see that some outliers are removed.

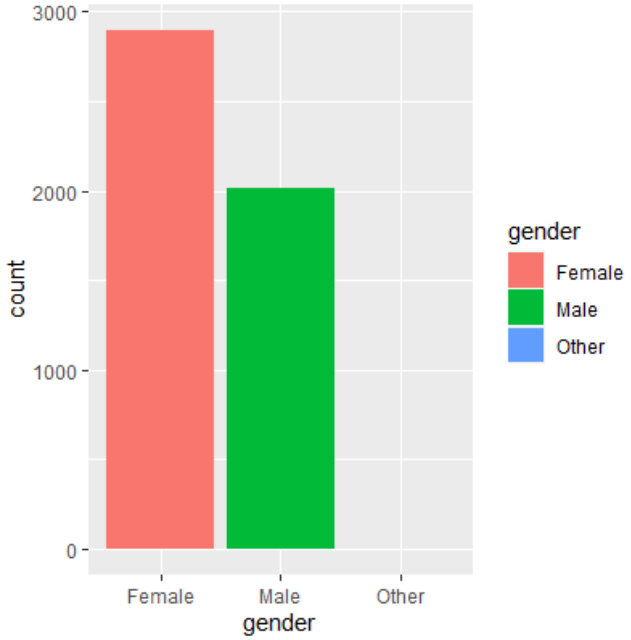
Presence of class imbalance might be there

```
> sum(df$stroke == "No")
[1] 4700
> sum(df$stroke == "Yes")
[1] 209
```

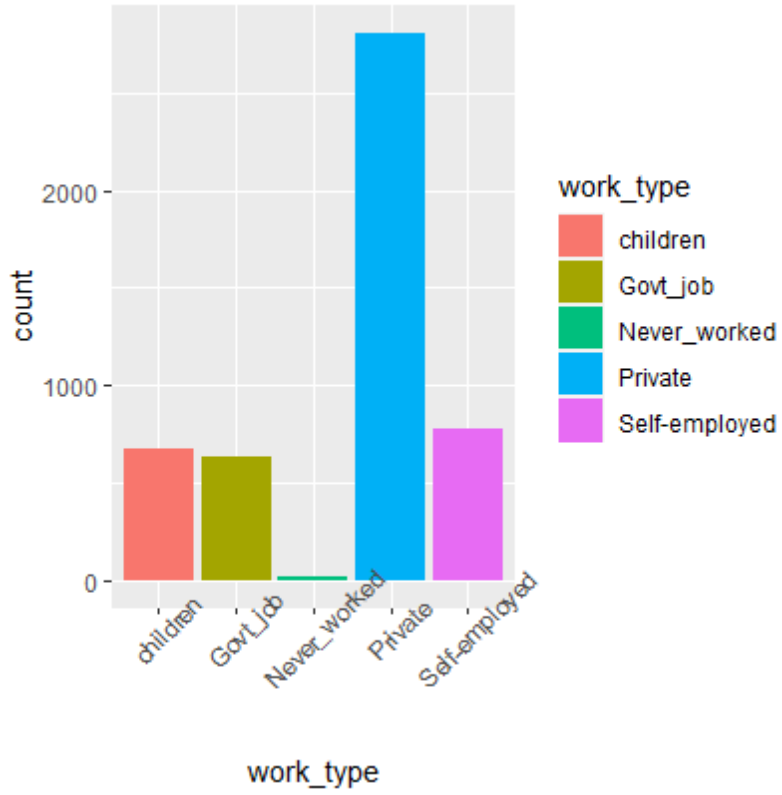
There are 4700 people with no stroke and 209 who had stroke.

b) Data Exploration

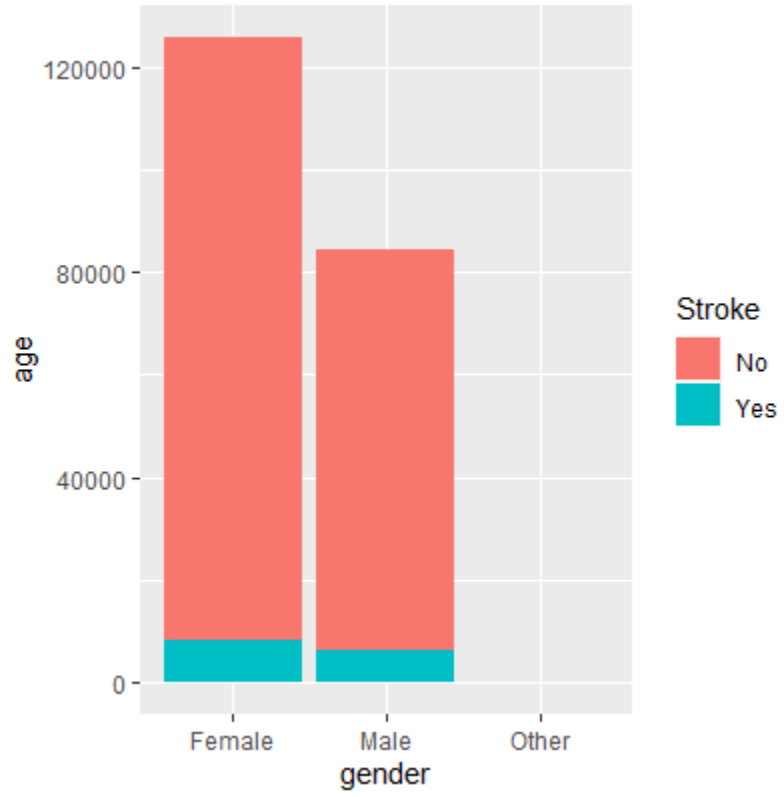
```
ggplot(data = df, aes(x=gender,fill=gender))+geom_bar()
```




```
ggplot(data = df, aes(x=work_type,fill=work_type))+geom_bar()+
  theme(axis.text.x =element_text(angle = 45) )
```



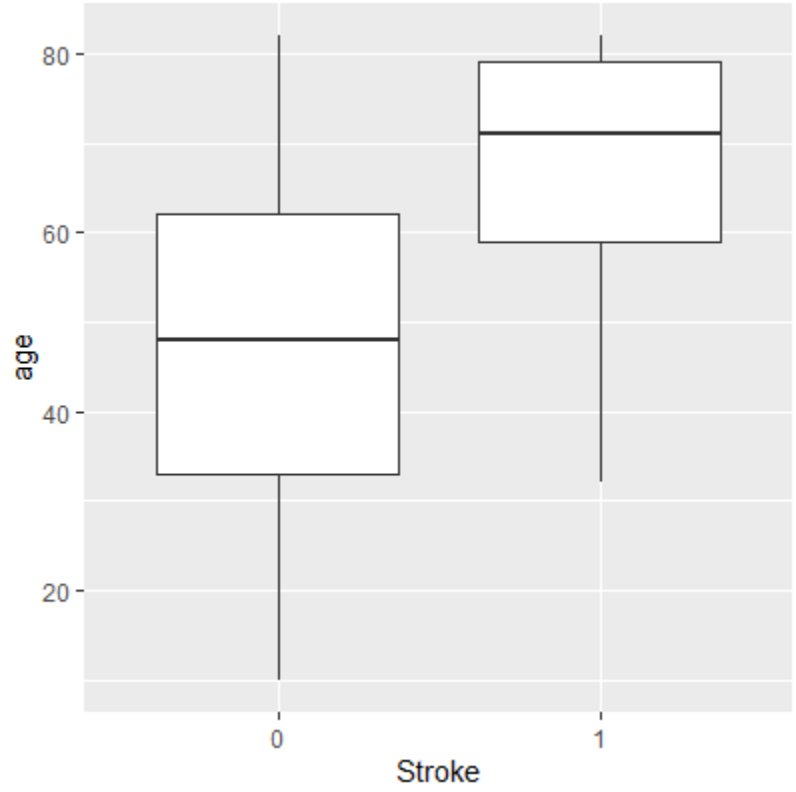
ggplot(df,aes(x=gender,y=age,fill=as.factor(stroke))) + geom_col() + labs(fill = "Stroke")



Summary of the dataset

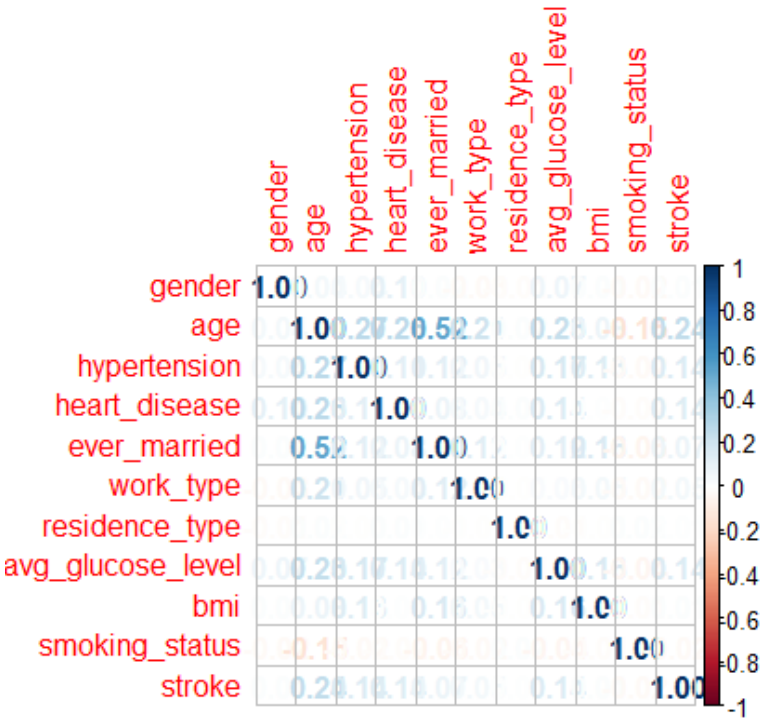
```
> summary(df)
       id          gender          age        hypertension heart_disease ever_married         work_type
 Min.   :   77   Female:2897   Min.   : 0.08   No :4458     No :4666      No :1705     children     : 671
 1st Qu.:18605   Male  :2011   1st Qu.:25.00   Yes: 451     Yes: 243      Yes:3204     Govt_job     : 630
 Median :37608   Other :   1   Median :44.00                                           Never_worked :  22
 Mean   :37064                 Mean   :42.87                                           Private      :2811
 3rd Qu.:55220                 3rd Qu.:60.00                                           Self-employed: 775
 Max.   :72940                 Max.   :82.00
 residence_type avg_glucose_level      bmi                 smoking_status stroke
 Rural:2419     Min.   : 55.12    Min.   :10.30   formerly smoked: 837   No :4700
 Urban:2490     1st Qu.: 77.07    1st Qu.:23.50   never smoked   :1852   Yes: 209
                Median : 91.68    Median :28.10   smokes         : 737
                Mean   :105.31    Mean   :28.89   Unknown        :1483
                3rd Qu.:113.57    3rd Qu.:33.10
                Max.   :271.74    Max.   :97.60
```

ggplot(df_num,aes(x=as.factor(stroke),y=age)) + geom_boxplot() + xlab("Stroke")



There is more chance of stroke for people after age of 59. People who get stroke have average age of 72, with maximum of 79

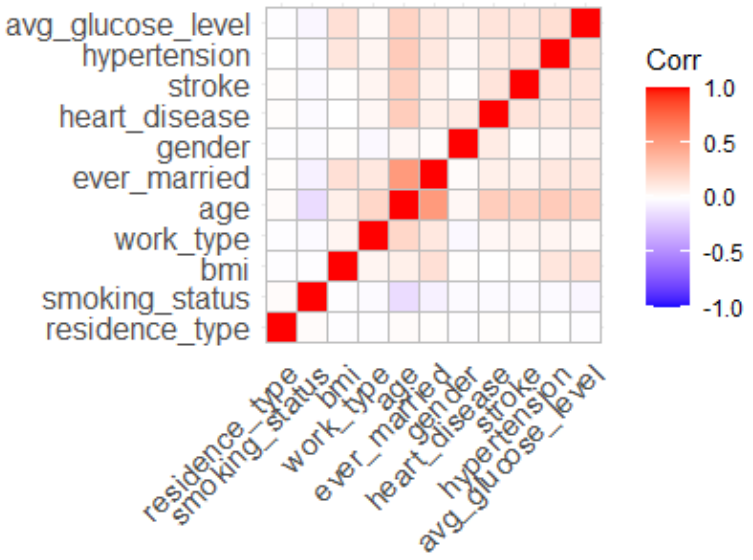
corrplot(cor(df_num),method = "number")



Correlation matrix with numeric values.

ggcorrplot(cor(df_num),hc.order = TRUE)

All the high correlated values are ordered one side.



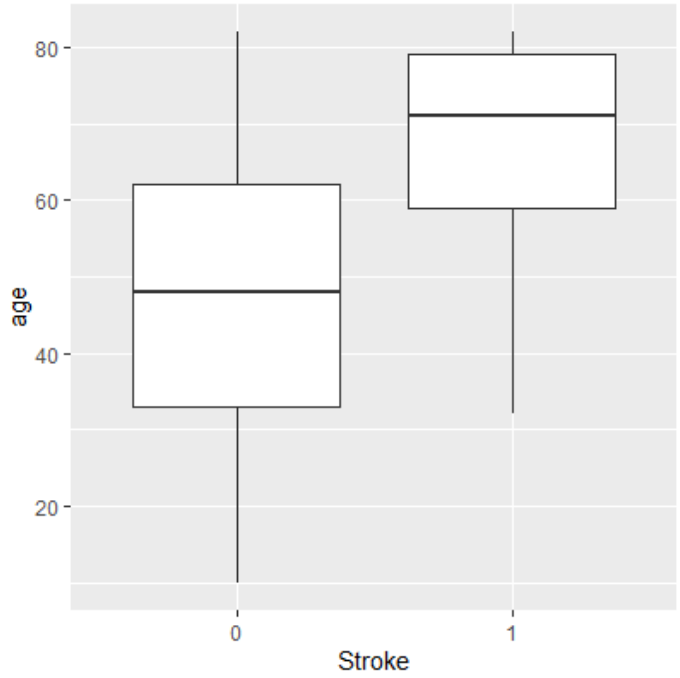
Heat map is easy for visualizing the correlation.

c) **Data Cleaning**

NA values, id column and Unknown smoking status will be removed. There are very few other labels for gender but it important to keep them to avoid bias and discrimination for ethics. There are many gender variants like transgender, non-binary, etc.

```
df_num <- df
df_num <- df_num[!(df_num$smoking_status == "Unknown"),]
df_num <- na.omit(df_num)
df_num <- subset(df_num,select = -id)
```

On page 1, the box plot had outliers which were removed later. Showing through visualization.



d) **Data Preprocessing**
To normalize the values, we will convert all categorical variables to numeric

```
df_num$gender <- ifelse(df_num$gender =="Male",1,0)
df_num$ever_married <- ifelse(df_num$ever_married == "Yes",1,0)
df_num$residence_type <- ifelse(df_num$residence_type == "Urban",1,0)
df_num$bmi <- as.numeric(df_num$bmi)
df_num$hypertension <- ifelse(df_num$hypertension == "Yes",1,0)
df_num$heart_disease <- ifelse(df_num$heart_disease == "Yes",1,0)
df_num$stroke <- ifelse(df_num$stroke == "Yes",1,0)

# private -> 4, self-employed -> 5, govt_job -> 2, children -> 1, never_worked -> 3
df_num$work_type <- as.integer(as.factor(df_num$work_type))
```

formerly smoked -> 1, never smoked -> 2, smokes -> 3
df_num$smoking_status <- as.integer(as.factor(df_num$smoking_status))

```
> glimpse(df_num)
Rows: 3,426
Columns: 11
$ gender            <dbl> 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1~
$ age               <dbl> 67, 80, 49, 79, 81, 74, 69, 81, 61, 54, 79, 50, 64, 75, 60, 71, 52, 79, 71, 80, 65~
$ hypertension      <dbl> 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0~
$ heart_disease     <dbl> 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0~
$ ever_married      <dbl> 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1~
$ work_type         <int> 4, 4, 4, 5, 4, 4, 4, 4, 2, 4, 4, 5, 4, 4, 4, 2, 5, 5, 4, 5, 4, 5, 4, 5, 5, 2, 4, 4~
$ residence_type    <dbl> 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0~
$ avg_glucose_level <dbl> 228.69, 105.92, 171.23, 174.12, 186.21, 70.09, 94.39, 80.43, 120.46, 104.51, 214.0~
$ bmi               <dbl> 36.6, 32.5, 34.4, 24.0, 29.0, 27.4, 22.8, 29.7, 36.8, 27.3, 28.2, 30.9, 37.5, 25.8~
$ smoking_status    <int> 1, 2, 3, 2, 1, 2, 2, 2, 3, 3, 2, 2, 3, 3, 2, 3, 2, 2, 1, 2, 1, 3, 3, 2, 1, 2, 1, 1~
$ stroke            <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
```

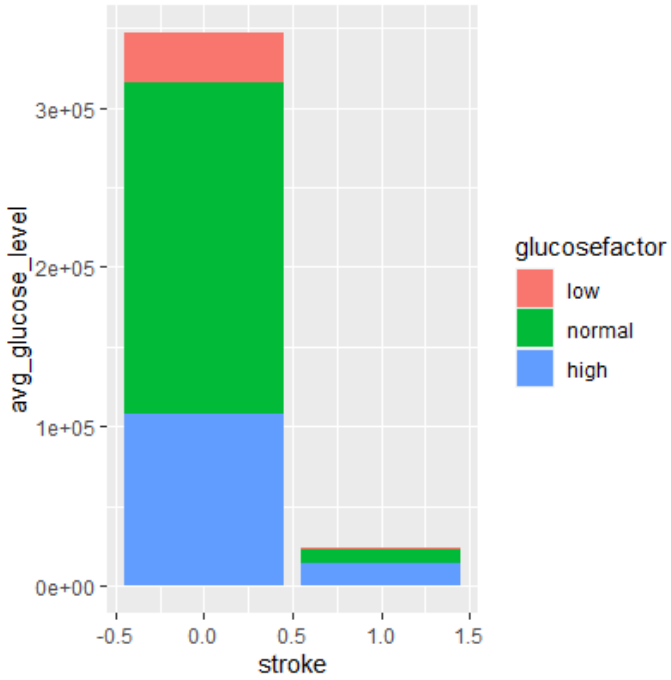
Next, we start binning and smoothing the glucose level to eliminate some noise,

```
df_num_bins <- df_num
df_num_bins <- df_num_bins %>% mutate(glucosefactor = cut(avg_glucose_level,
                                    breaks = c(-Inf,70,140,Inf),
                                    labels=c("low","normal","high")))

low <- df_num_bins %>%
  filter(glucosefactor == 'low') %>%
  mutate(avg_glucose_level = mean(avg_glucose_level, na.rm = T))
normal <- df_num_bins %>%
  filter(glucosefactor == 'normal') %>%
  mutate(avg_glucose_level = mean(avg_glucose_level, na.rm = T))
high <- df_num_bins %>%
  filter(glucosefactor == 'high') %>%
  mutate(avg_glucose_level = mean(avg_glucose_level, na.rm = T))
df_num_bins <- bind_rows(list(low, normal, high))
```

The new values were mean of individual glucose factor.

```
> head(df_num_bins)
  gender age hypertension heart_disease ever_married work_type residence_type avg_glucose_level  bmi
1      0  49            0             0            1         4              1          63.18738 29.9
2      0  39            1             0            1         4              0          63.18738 39.2
3      0  82            0             0            1         4              0          63.18738 33.2
4      0  67            1             0            1         5              0          63.18738 25.3
5      0  80            0             1            1         5              0          63.18738 21.7
6      0  70            0             1            1         4              0          63.18738 32.3
  smoking_status stroke glucosefactor
1              2      1           low
2              3      1           low
3              2      1           low
4              3      1           low
5              1      1           low
6              1      1           low
```

```
> summary(df_min_max)
     gender            age          hypertension    heart_disease      ever_married      work_type
 Min.   :0.0000   Min.   :10.00   Min.   :0.0000   Min.   :0.00000   Min.   :0.0000   Min.   :1.00
 1st Qu.:0.0000   1st Qu.:34.00   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:1.0000   1st Qu.:4.00
 Median :0.0000   Median :50.00   Median :0.0000   Median :0.00000   Median :1.0000   Median :4.00
 Mean   :0.3908   Mean   :48.65   Mean   :0.1191   Mean   :0.06013   Mean   :0.7586   Mean   :3.82
 3rd Qu.:1.0000   3rd Qu.:63.00   3rd Qu.:0.0000   3rd Qu.:0.00000   3rd Qu.:1.0000   3rd Qu.:4.00
 Max.   :1.0000   Max.   :82.00   Max.   :1.0000   Max.   :1.00000   Max.   :1.0000   Max.   :5.00
 residence_type   avg_glucose_level       bmi.bmi       smoking_status      stroke         glucosefactor
 Min.   :0.0000   Min.   : 63.19    Min.   :0.0000000   Min.   :1.000   Min.   :0.00000   low    : 508
 1st Qu.:0.0000   1st Qu.: 94.02    1st Qu.:0.1714286   1st Qu.:2.000   1st Qu.:0.00000   normal:2304
 Median :1.0000   Median : 94.02    Median :0.2186335   Median :2.000   Median :0.00000   high   : 614
 Mean   :0.5093   Mean   :108.32    Mean   :0.2334167   Mean   :1.971   Mean   :0.05254
 3rd Qu.:1.0000   3rd Qu.: 94.02    3rd Qu.:0.2807453   3rd Qu.:2.000   3rd Qu.:0.00000
 Max.   :1.0000   Max.   :199.33    Max.   :1.0000000   Max.   :3.000   Max.   :1.00000
```



Min-max normalization is one of the most common ways to normalize data. For every feature, the minimum value of that feature gets transformed into a 0, the maximum value gets transformed into a 1.

```
normalize = function(x){
  return((x-min(x,na.rm = TRUE))/(max(x,na.rm = TRUE)-min(x,na.rm = TRUE)))
}

# 2 is for applying all rows and 1 for all columns
df_min_max <- df_num_bins
df_min_max$bmi <- as.data.frame(apply(df_num_bins[9],2, normalize))
```

```
> head(df_min_max)
  gender age hypertension heart_disease ever_married work_type residence_type avg_glucose_level       bmi
1      0  49            0             0            1         4              1          63.18738 0.2285714
2      0  39            1             0            1         4              0          63.18738 0.3440994
3      0  82            0             0            1         4              0          63.18738 0.2695652
4      0  67            1             0            1         5              0          63.18738 0.1714286
5      0  80            0             1            1         5              0          63.18738 0.1267081
6      0  70            0             1            1         4              0          63.18738 0.2583851
  smoking_status stroke glucosefactor
1              2      1           low
2              3      1           low
3              2      1           low
4              3      1           low
5              1      1           low
6              1      1           low
```

We can see that bmi (body mass index) column has been now normalized i.e., all values are now in between 0 and 1.

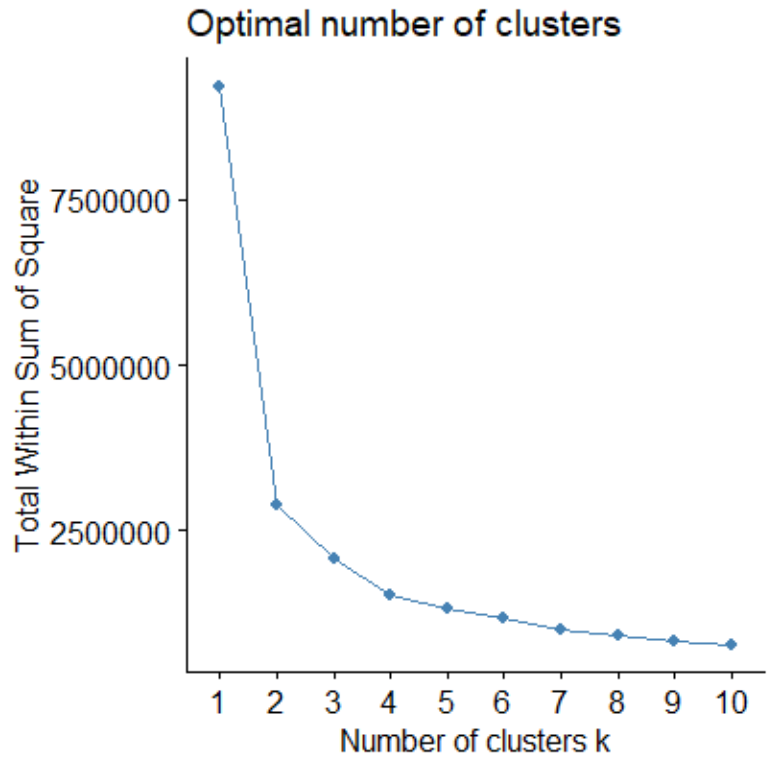
e) **Clustering**

I used k-means clustering to find groups which have not been explicitly labeled in the data. To do this I removed stroke column from the dataset.

df_num <- subset(df_num,select = -stroke)

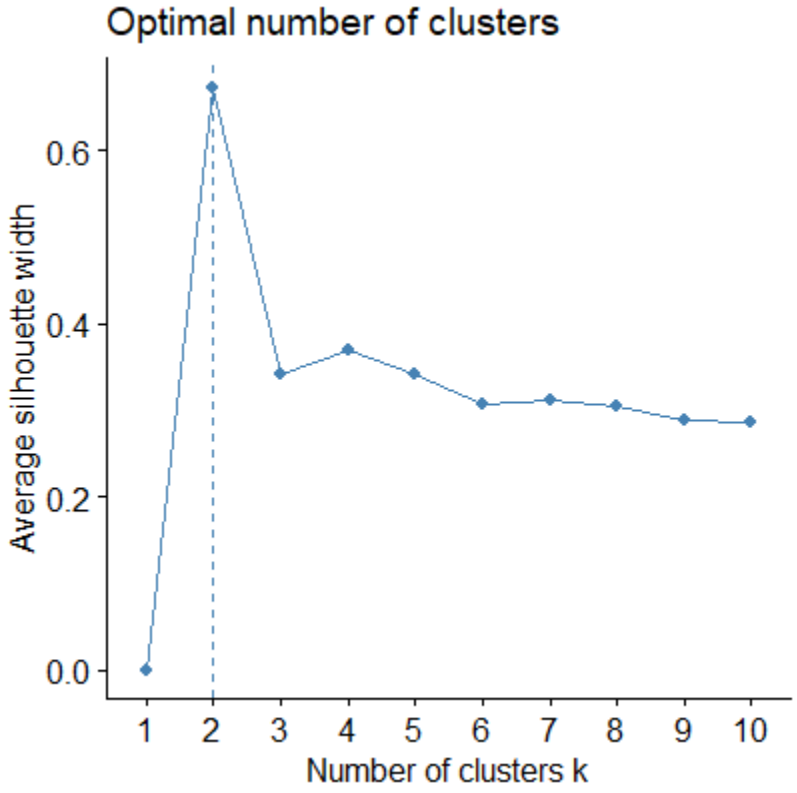
We will use optimal number of clusters with knee plot and silhouette.

fviz_nbclust(df_num, kmeans, method = "wss") # 3 or 4

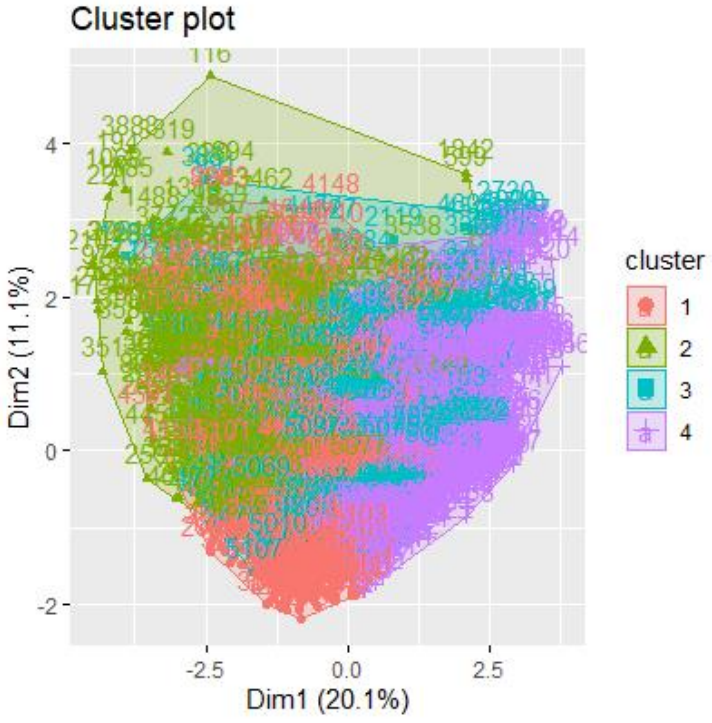


Knee plot suggests 4 clusters

```
fviz_nbclust(df_num, kmeans, method = "silhouette")
```

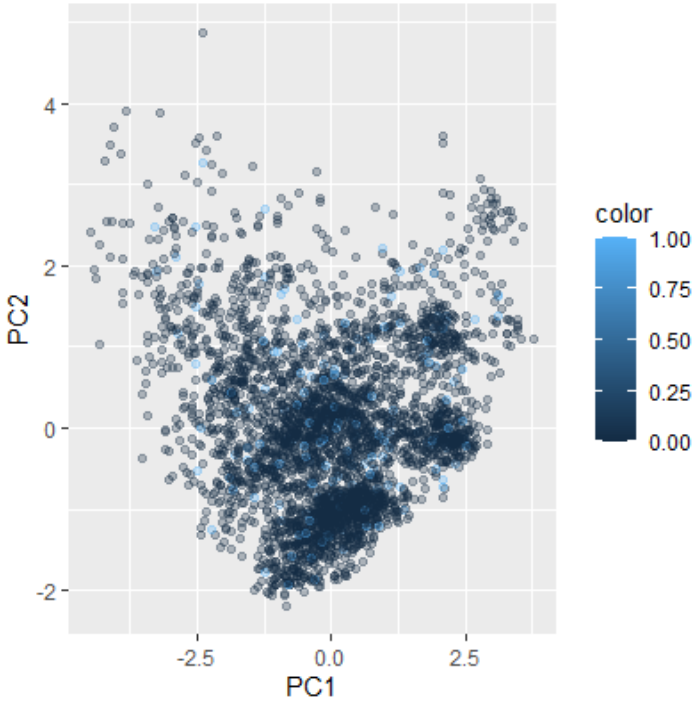



```
fit <- kmeans(df_num, centers = 4, nstart = 25)
fit
fviz_cluster(fit, data = df_num)
```



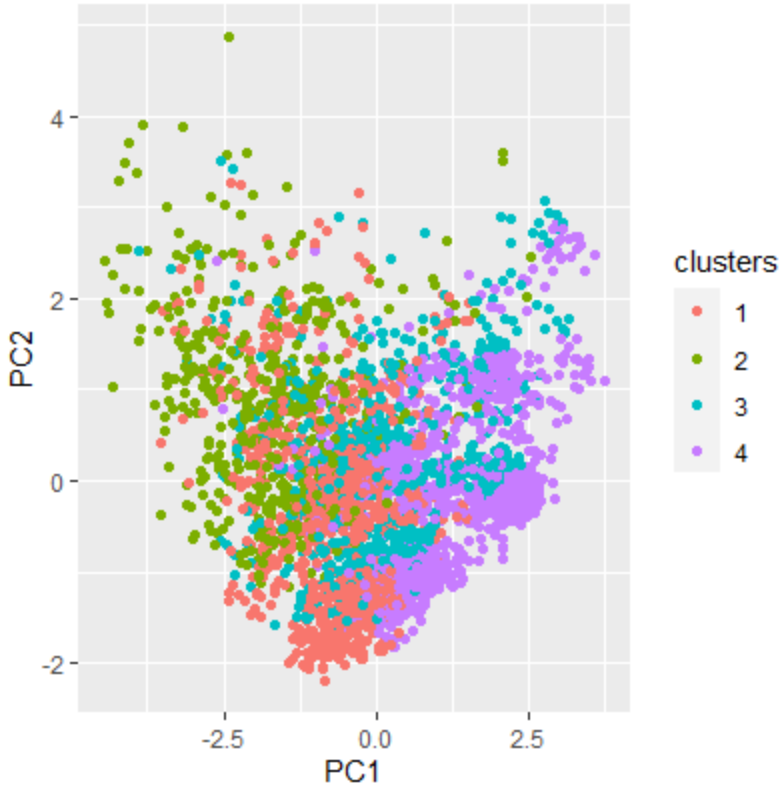
For comparison we can generate our own PCA plot and color the points based on their respective labels.

```
preproc <- preProcess(df_num, method=c("center", "scale"))
df_num <- predict(preproc, df_num)
pca = prcomp(df_num)
rotated_data = as.data.frame(pca$x)
rotated_data$color <- df_num_bins$stroke
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = color)) + geom_point(alpha = 0.3)
```

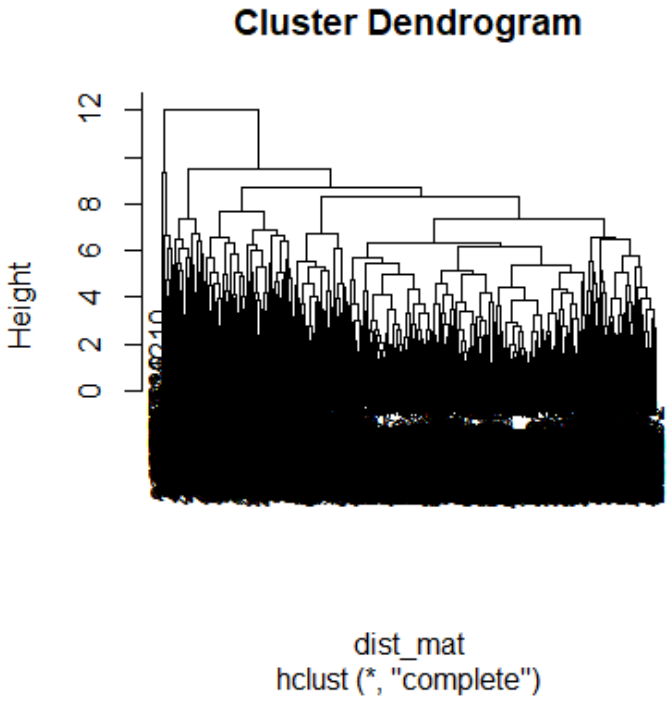


After assigning clusters to the rotated data set, we can color for individual points and get rid of the area markers

```
rotated_data$clusters = as.factor(fit$cluster)
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = clusters)) + geom_point()
```



Next, we use agglomerative clustering the most common type of hierarchical clustering used to group objects in clusters based on their similarity. Next, pairs of clusters are successively merged until all clusters have been merged into one big cluster.



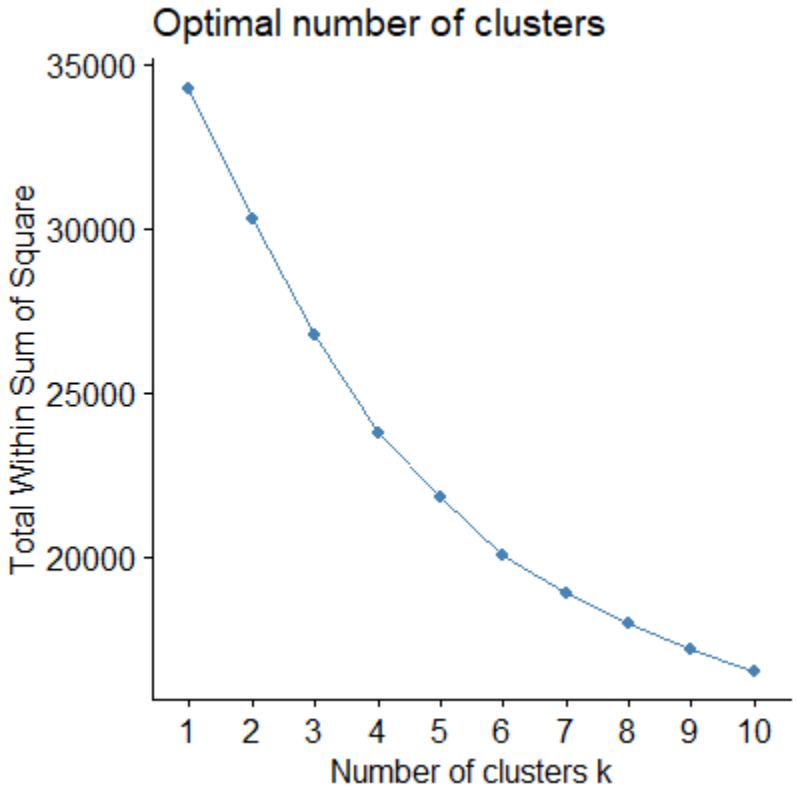
```
df_num_gower <- df_num_bins
df_num_gower <- subset(df_num_bins,select = -glucosefactor)
df_num_gower$stroke <- ifelse(df_num_gower$stroke ==1,"Yes","No")
dist_mat2 <- daisy(df_num, metric = "gower")
summary(dist_mat2)
```

```
> summary(dist_mat2)
5867025 dissimilarities, summarized :
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
0.0006916 0.2012700 0.2819700 0.2867900 0.3653900 0.8637400
Metric :  mixed ;  Types = I, I, I, I, I, I, I, I, I, I
Number of objects : 3426
```

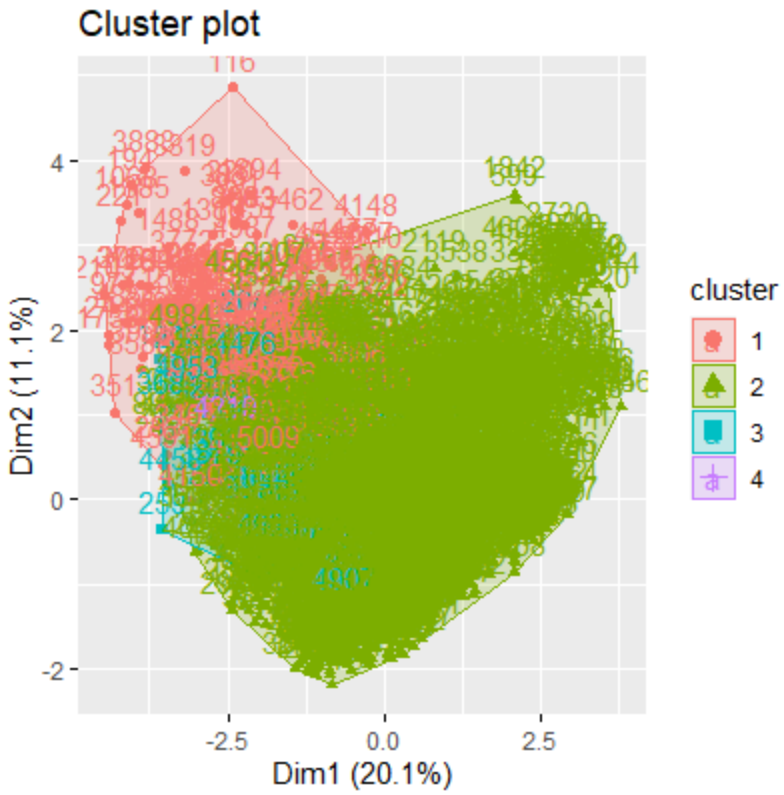
df_num version creates a 47.2 MB dissimilarity matrix

To move on with the HAC we need cutoff point determining number of clusters.

```
fviz_nbclust(df_num, FUN = hcut, method = "wss")
```




```
fviz_nbclust(df_num, FUN = hcut, method = "silhouette")
rotated_data$color <- ifelse(rotated_data$color =="1","Yes","No")
h4 <- cutree(hfit, k=4)
fviz_cluster(list(data = df_num, cluster = h4))
```

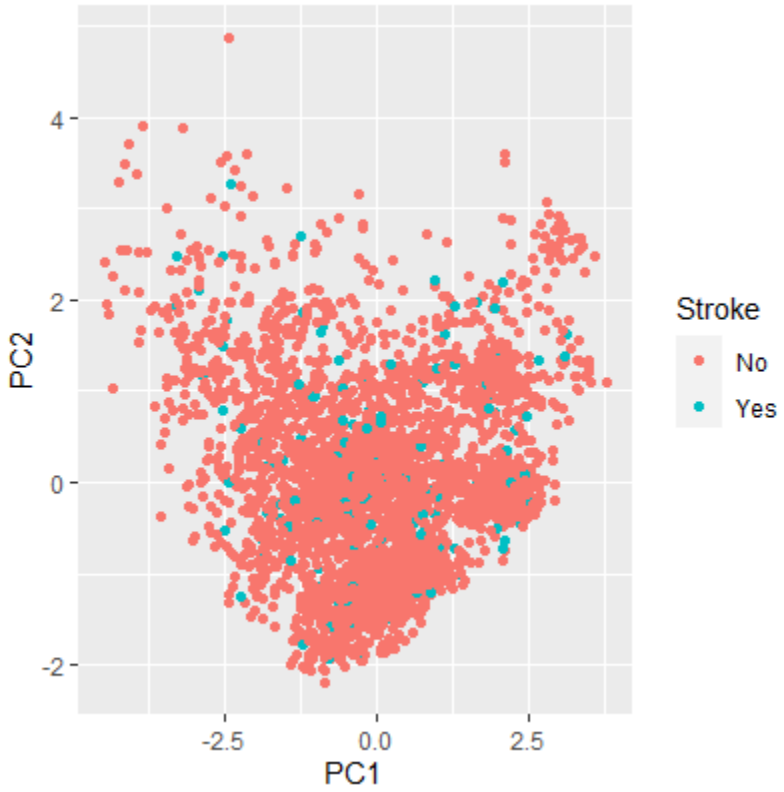


Point visualization

ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = clusters)) + geom_point()

Class labels on 2-D plane

```
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = color)) + geom_point()+
  scale_colour_discrete("Stroke")
```



To compare the models, we can also create a table of predictions

```
result <- data.frame(Stroke = rotated_data$color, HAC4 = h4, Kmeans = fit$cluster)
head(result, n = 20)
```

```
> head(result, n = 20)
   Stroke HAC4 Kmeans
1     Yes    1      2
3     Yes    1      1
4     Yes    2      2
5     Yes    2      2
6     Yes    2      2
7     Yes    1      1
8     Yes    2      1
11    Yes    2      1
12    Yes    1      3
13    Yes    2      3
15    Yes    1      2
16    Yes    2      2
17    Yes    1      2
18    Yes    2      2
19    Yes    2      1
21    Yes    2      2
22     No    3      2
23     No    2      2
25     No    2      1
26     No    2      1
```

Now, we will create cross tabulation for HAC and k-means

result %>% group_by(HAC4) %>% select(HAC4, Stroke) %>% table()
result %>% group_by(Kmeans) %>% select(Kmeans, Stroke) %>% table()

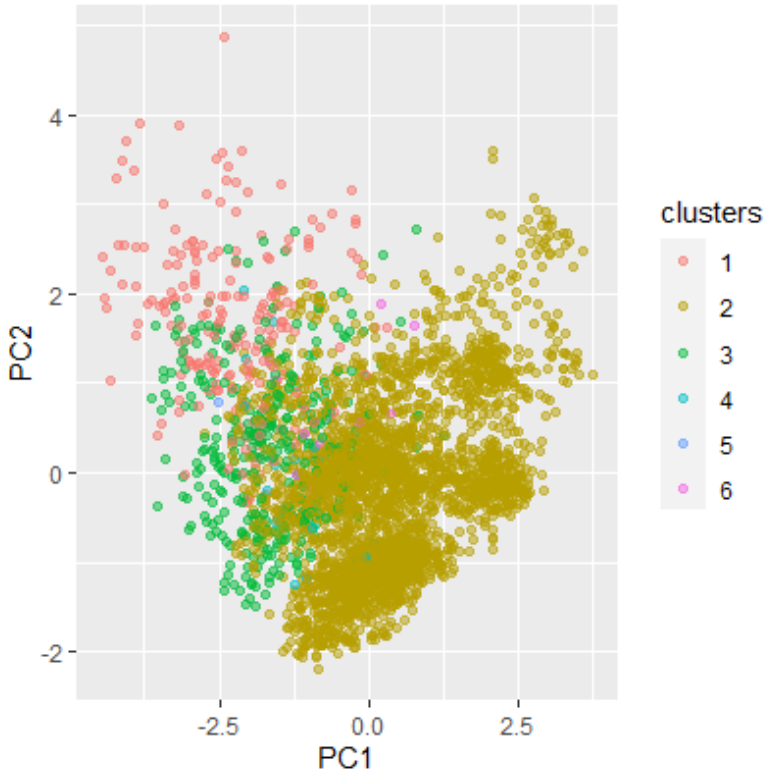
```
> result %>% group_by(HAC4) %>% select(HAC4, Stroke) %>% table()
    Stroke
HAC4   No  Yes
   1  196   10
   2 2971  167
   3   77    2
   4    2    1
> # cross tab for k means
> result %>% group_by(Kmeans) %>% select(Kmeans, Stroke) %>% table()
      Stroke
Kmeans   No  Yes
     1 1016   56
     2  465   24
     3  701   40
     4 1064   60
```

the results of HAC and k-means are very similar, we can say that k-means is performing slightly better because in the 4th cluster data is better clustered than HAC.

At higher number of clusters this metric usually plays a more crucial role we try with 6 clusters and let's try average, median, centroid, and single linage metrics.

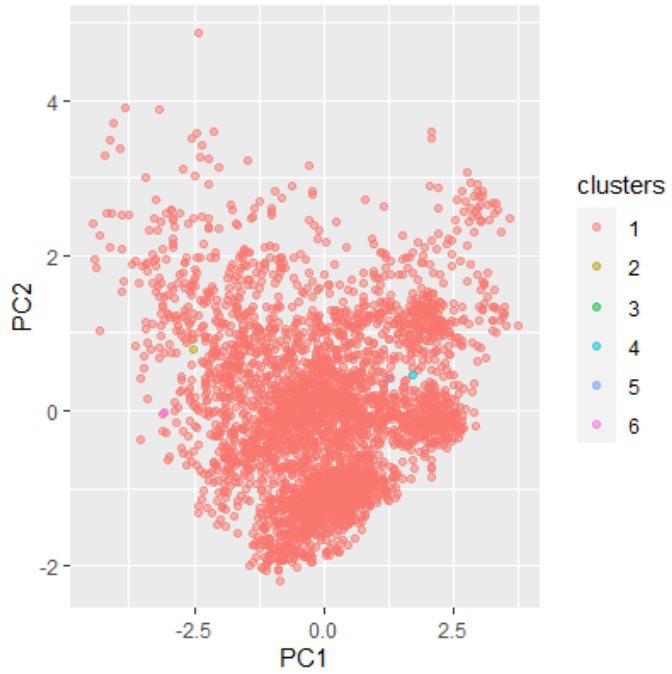
Average - The distance between two clusters is the distance between mean of the elements in cluster 1 and the mean of the elements in cluster 2.

hfit <- hclust(dist_mat, method = 'average')
h6 <- cutree(hfit, k=6)
rotated_data$clusters = as.factor(h6)
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = clusters)) + geom_point(alpha = 0.5)
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = color)) + geom_point(alpha = 0.5)



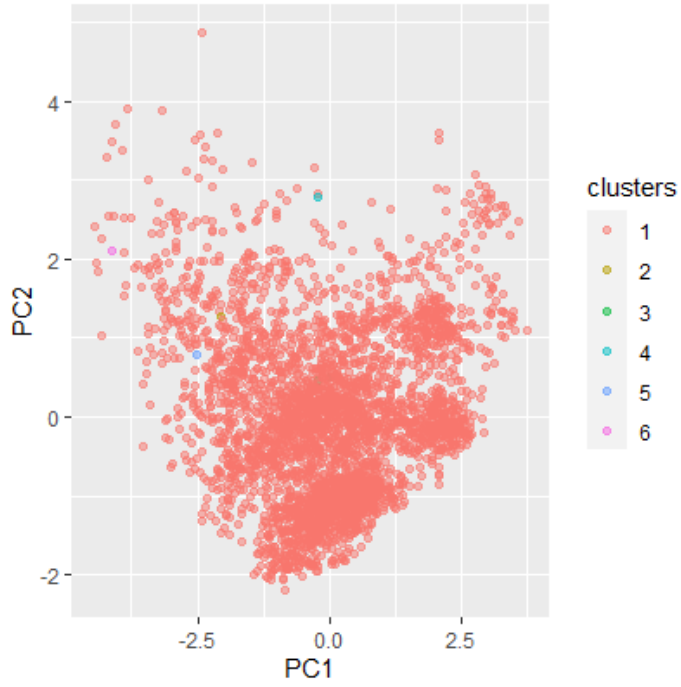
Median - The distance between two clusters is the distance between the median of elements in cluster 1 and the median of elements in cluster 2.

```
hfit <- hclust(dist_mat, method = 'median')
h6 <- cutree(hfit, k=6)
rotated_data$clusters = as.factor(h6)
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = clusters)) + geom_point(alpha = 0.5)
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = color)) + geom_point(alpha = 0.5)
```



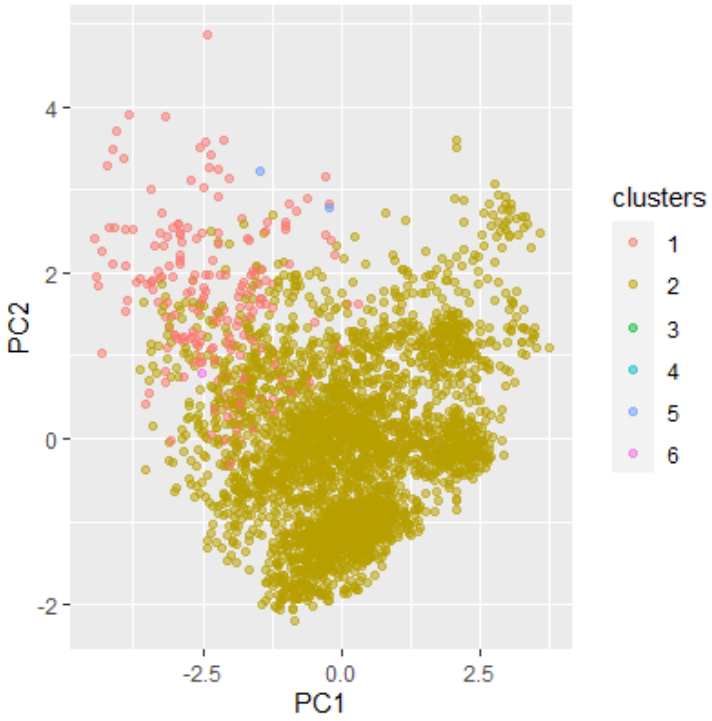
The centroid version is using the distance between cluster centroids. The distance between two clusters is the distance between the centroid for cluster 1 and the centroid for cluster 2.

```
hfit <- hclust(dist_mat, method = 'centroid')
h6 <- cutree(hfit, k=6)
rotated_data$clusters = as.factor(h6)
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = clusters)) + geom_point(alpha = 0.5)
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = color)) + geom_point(alpha = 0.5)
```



The single linkage method adopts a 'friends of friends' clustering strategy according to the hclust function documentation. The distance between two clusters is defined as the minimum value of all pairwise distances for the elements in cluster 1 and the elements in cluster 2.

```
hfit <- hclust(dist_mat, method = 'single')
h6 <- cutree(hfit, k=6)
rotated_data$clusters = as.factor(h6)
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = clusters)) + geom_point(alpha = 0.5)
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = color)) + geom_point(alpha = 0.5)
```



For every distance metric clustering will be different because of the calculation to set the data points.

f) **Classification**

```
stroke1 <- glm(stroke~ gender+ age + hypertension + avg_glucose_level + smoking_status + bmi,
          data=df_num, family = binomial(link = "logit"))

stroke2<- glm(stroke~ age + hypertension + avg_glucose_level + bmi ,
          data=df_num, family = binomial(link = "logit"))

stroke3<- glm(stroke~ age + heart_disease + work_type + residence_type + avg_glucose_level + bmi,
          data=df_num, family = binomial(link = "logit"))

stroke4<- glm(stroke~ age + heart_disease + avg_glucose_level ,
          data=df_num, family = binomial(link = "logit"))
```

```
stroke5<- glm(stroke~ age + ever_married + avg_glucose_level ,
          data=df_num, family = binomial(link = "logit"))
summary(stroke1)
summary(stroke2)
summary(stroke3)
summary(stroke4)
summary(stroke5)
```

```
> summary(stroke1)

Call:
glm(formula = stroke ~ gender + age + hypertension + avg_glucose_level +
    smoking_status + bmi, family = binomial(link = "logit"),
    data = df_num)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0715  -0.3425  -0.1962  -0.1043   3.1714

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)       -8.324736   0.704151 -11.822  < 2e-16 ***
gender            -0.001356   0.164152  -0.008 0.993409
age                0.071961   0.006543  10.998  < 2e-16 ***
hypertension       0.568319   0.181771   3.127 0.001769 **
avg_glucose_level  0.004863   0.001365   3.563 0.000367 ***
smoking_status     0.151906   0.118421   1.283 0.199576
bmi                0.005814   0.012792   0.455 0.649461
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1411.0  on 3425  degrees of freedom
Residual deviance: 1152.2  on 3419  degrees of freedom
AIC: 1166.2

Number of Fisher Scoring iterations: 7
```

```
> summary(stroke2)

Call:
glm(formula = stroke ~ age + hypertension + avg_glucose_level +
    bmi, family = binomial(link = "logit"), data = df_num)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0544  -0.3397  -0.1940  -0.1047   3.2100

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)       -7.933806   0.624203 -12.710  < 2e-16 ***
age                0.070697   0.006453  10.956  < 2e-16 ***
hypertension       0.571130   0.181765   3.142 0.001677 **
avg_glucose_level  0.004803   0.001361   3.529 0.000417 ***
bmi                0.005335   0.012777   0.418 0.676307
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1411.0  on 3425  degrees of freedom
Residual deviance: 1153.9  on 3421  degrees of freedom
AIC: 1163.9

Number of Fisher Scoring iterations: 7

> summary(stroke3)

Call:
glm(formula = stroke ~ age + heart_disease + work_type + residence_type +
    avg_glucose_level + bmi, family = binomial(link = "logit"),
    data = df_num)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0333  -0.3391  -0.1954  -0.1059   3.2033

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)       -8.002492   0.681862 -11.736  < 2e-16 ***
age                0.071615   0.006579  10.886  < 2e-16 ***
heart_disease      0.480541   0.215615   2.229 0.025834 *
work_type         -0.024804   0.085710  -0.289 0.772282
residence_type     0.010457   0.161009   0.065 0.948214
avg_glucose_level  0.004906   0.001359   3.609 0.000307 ***
bmi                0.010431   0.012793   0.815 0.414840
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1411.0  on 3425  degrees of freedom
Residual deviance: 1158.5  on 3419  degrees of freedom
AIC: 1172.5

Number of Fisher Scoring iterations: 7
```

```
> summary(stroke4)

Call:
glm(formula = stroke ~ age + heart_disease + avg_glucose_level,
    family = binomial(link = "logit"), data = df_num)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0523  -0.3374  -0.1962  -0.1083   3.1891

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)       -7.721026   0.436686 -17.681  < 2e-16 ***
age                0.070246   0.006275  11.194  < 2e-16 ***
heart_disease      0.473941   0.215660   2.198    0.028 *
avg_glucose_level  0.005177   0.001321   3.918 8.92e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1411.0  on 3425  degrees of freedom
Residual deviance: 1159.2  on 3422  degrees of freedom
AIC: 1167.2

Number of Fisher Scoring iterations: 7

> summary(stroke5)

Call:
glm(formula = stroke ~ age + ever_married + avg_glucose_level,
    family = binomial(link = "logit"), data = df_num)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.9592  -0.3407  -0.1963  -0.1089   3.2083

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)       -7.716135   0.466366 -16.545  < 2e-16 ***
age                0.073145   0.006163  11.869  < 2e-16 ***
ever_married      -0.182112   0.259603  -0.702    0.483
avg_glucose_level  0.005475   0.001314   4.167 3.08e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1411.0  on 3425  degrees of freedom
Residual deviance: 1163.3  on 3422  degrees of freedom
AIC: 1171.3

Number of Fisher Scoring iterations: 7
```

Use 5-fold cross validation to assess models' quality for model 2

```
model21<- glm(stroke~ age + avg_glucose_level + bmi, data=df_num,
        family = binomial(link = "logit"))
model22<- glm(stroke~ age * hypertension + avg_glucose_level + bmi , data=df_num,
        family = binomial(link = "logit"))
model23<- glm(stroke~ age + hypertension * avg_glucose_level + bmi, data=df_num,
        family = binomial(link = "logit"))
model24<- glm(stroke~ age + gender * hypertension + avg_glucose_level + bmi, data=df_num,
        family = binomial(link = "logit"))
model25<- glm(stroke~ age + hypertension * bmi + avg_glucose_level, data=df_num,
        family = binomial(link = "logit"))
```

AIC (Akaike information criterion) is a mathematical method for evaluating how well a model fits the data it was generated from AIC is lower for model 2 among the 5 models performed.

```
aic_values <- AIC(model21, model22, model23, model24, model25)
aic_values
```

```
> aic_values
        df      AIC
model21  4 1171.279
model22  6 1163.980
model23  6 1165.806
model24  7 1167.281
model25  6 1164.870
```

When comparing model fitted by maximum likelihood to the same data, the smaller the AIC, the better the fit.
Model 22 is the best one.

```
summary(model22)
```

```
> summary(model22)

Call:
glm(formula = stroke ~ age * hypertension + avg_glucose_level +
    bmi, family = binomial(link = "logit"), data = df_num)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.9651  -0.3464  -0.1883  -0.0978   3.2529

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)       -8.140618   0.653403 -12.459  < 2e-16 ***
age                0.074818   0.007256  10.312  < 2e-16 ***
hypertension       2.055729   1.058539   1.942  0.05213 .
avg_glucose_level  0.004792   0.001356   3.534  0.00041 ***
bmi                0.003211   0.012851   0.250  0.80269
age:hypertension  -0.021568   0.015275  -1.412  0.15796
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1411  on 3425  degrees of freedom
Residual deviance: 1152  on 3420  degrees of freedom
AIC: 1164

Number of Fisher Scoring iterations: 7
```

Three predictors can reject null hypothesis.

Decision Tree
Splitting the dataset in 70:30. 70% for training set and 30% for test set.

```
df_num$stroke <- ifelse(df_num$stroke == 1,"Yes","No")
df_num$stroke <- as.factor(df_num$stroke)

set.seed(372)
index = createDataPartition(y=df_num$stroke, p=0.7, list=FALSE)
train_set = df_num[index,]
test_set = df_num[-index,]
train_control = trainControl(method = "cv", number = 10)
```

setting up hyperparameters,
Minsplit is the minimum number of observations that must exist in a node in order for a split to be attempted.
Maxdepth - Controls the maximum depth of the tree that will be created.
minbucket provides the smallest number of observations that are allowed in a terminal node

```
hypers = rpart.control(minsplit =  500, maxdepth = 3, minbucket = 2000)
tree1 <- train(stroke ~., data = train_set, method = "rpart1SE", trControl = train_control)
tree1
```

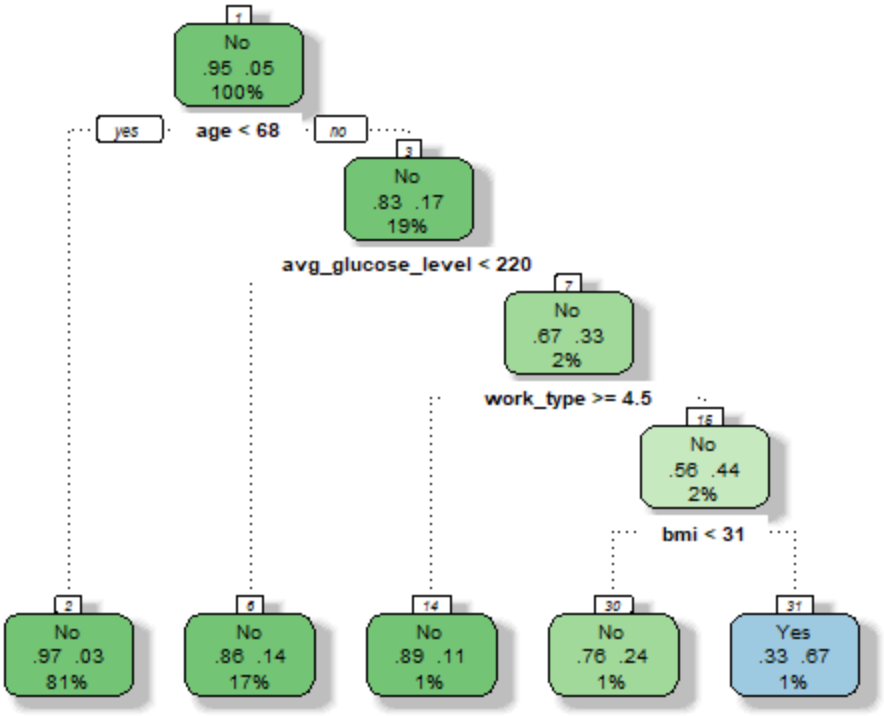
```
> tree1
CART

2399 samples
  10 predictor
   2 classes: 'No', 'Yes'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 2158, 2160, 2160, 2159, 2159, 2159, ...
Resampling results:

  Accuracy   Kappa
  0.9449824  -0.00458173
```

Using the training set we build our model and make predictions on the test set for evaluations

pred_tree <- predict(tree1, test_set)
cm <- confusionMatrix(as.factor(test_set$stroke), pred_tree)
cm

```
> cm
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  971   2
       Yes  53   1

               Accuracy : 0.9464
                 95% CI : (0.9309, 0.9594)
    No Information Rate : 0.9971
    P-Value [Acc > NIR] : 1

                  Kappa : 0.0297

 Mcnemar's Test P-Value : 1.562e-11

            Sensitivity : 0.94824
            Specificity : 0.33333
         Pos Pred Value : 0.99794
         Neg Pred Value : 0.01852
             Prevalence : 0.99708
         Detection Rate : 0.94547
   Detection Prevalence : 0.94742
      Balanced Accuracy : 0.64079

       'Positive' Class : No
```

In the g) section we will interpret the classification table.



If you age < 68 it is predicting no stroke, if greater 68 it checks glucose level. If avg_glucose_level < 220 no stroke and greater than 220 it checks work type. If a person has private job or self-employed he has chance of getting a stroke. Next we check bmi for those two work type and if bmi < 31, no stroke or else the person had a stroke

g) **Evaluation**
Not binning the class as it has 2 classes.

(1) Produce 2x2 confusion matrix

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  971   2
       Yes  53   1

               Accuracy : 0.9464
                 95% CI : (0.9309, 0.9594)
    No Information Rate : 0.9971
    P-Value [Acc > NIR] : 1

                  Kappa : 0.0297

 Mcnemar's Test P-Value : 1.562e-11

            Sensitivity : 0.94824
            Specificity : 0.33333
         Pos Pred Value : 0.99794
         Neg Pred Value : 0.01852
             Prevalence : 0.99708
         Detection Rate : 0.94547
   Detection Prevalence : 0.94742
      Balanced Accuracy : 0.64079

       'Positive' Class : No
```

(2) Calculating precision and recall .

Precision = TP/(TP+FP) → 0.18
recall = TP/(TP+FN) → 0.333

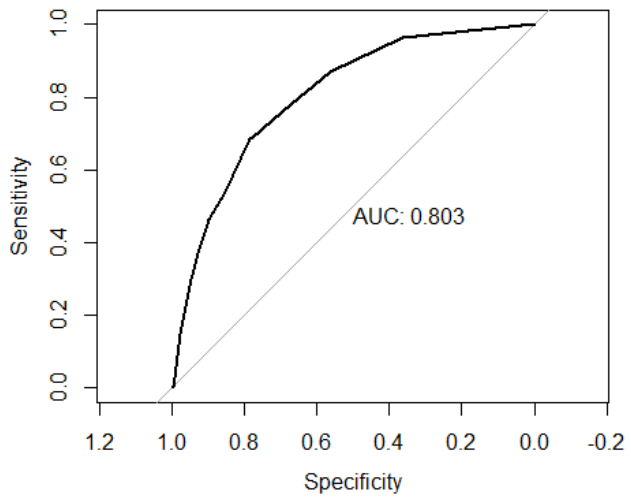
(3) ROC
Probabilities of belonging to a certain class.

knn <- train(stroke ~., data = train_set, method = "knn", trControl = train_control, tuneLength = 20)
knn

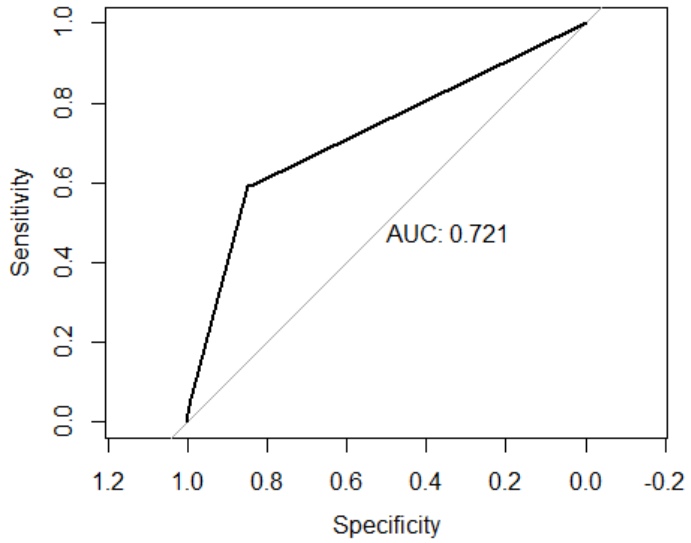
```
> knn
k-Nearest Neighbors

2399 samples
  10 predictor
   2 classes: 'No', 'Yes'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 2158, 2160, 2160, 2160, 2158, 2159, ...
Resampling results across tuning parameters:

  k   Accuracy   Kappa
   5  0.9416486   0.039778055
   7  0.9441521  -0.006038411
   9  0.9462389  -0.002236027
  11  0.9470688  -0.000776583
  13  0.9474837   0.000000000
  15  0.9474837   0.000000000
  17  0.9474837   0.000000000
  19  0.9474837   0.000000000
  21  0.9474837   0.000000000
  23  0.9474837   0.000000000
  25  0.9474837   0.000000000
  27  0.9474837   0.000000000
  29  0.9474837   0.000000000
  31  0.9474837   0.000000000
  33  0.9474837   0.000000000
  35  0.9474837   0.000000000
  37  0.9474837   0.000000000
  39  0.9474837   0.000000000
  41  0.9474837   0.000000000
  43  0.9474837   0.000000000

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 43.
```

```
pred_prob <- predict(knn, test_set, type = "prob")
roc_obj <- roc((test_set$stroke), pred_prob[,1])
plot(roc_obj, print.auc=TRUE)
```

AUC provides an aggregate measure of performance across all possible classification thresholds.



```
tree1 <- train(stroke ~., data = train_set, method = "rpart1SE", trControl = train_control)
roc_obj2 <- roc((test_set$stroke), pred_prob2[,1])
plot(roc_obj2, print.auc=TRUE)
```



ROC curves can sometimes be misleading in some very imbalanced applications. AUC metric on the other hand, combines the scores of sensitivity and specificity and it gives you a middle ground metric.

Here we have higher AUC and balanced accuracy score for knn. Higher the AUC, the better the model is at predicting 0s as 0s and 1s as 1s. By analogy, the Higher the AUC, the better the model is at distinguishing between patients with no stroke and stroke.

h) **Report**

In the given dataset we used k-means and logistic regression to forecast whether the patient can have stroke or not. We had to deal with imbalanced data which is common in such healthcare problems. We removed the rows with empty cells (NA values) and for smoking status we removed the unknown labels as it was considered as missing data. But it could have in other way like Age less than 10 or 15 years patients could have been tagged as never smoked etc. We saw that age, BMI, and glucose level are the most important features when it comes to predicting stroke-prone individuals, based on the current dataset. It was also observed that women are prone to stroke on average at a much older age in comparison to males which experiences strokes on average as soon. We saw that k-means, logistic regression and decision tree were trained on an unsampled version of the original dataset yielded satisfiable results and the accuracy of the original model was improved with 10-fold cross validation.

I) **Reflection**

This course has been very important for me as the whole data pipeline was explained. The EDA, cleaning, preprocessing, clustering, classification, and some advanced evaluation techniques, this whole process is very crucial to become a data scientist. Many machine learning algorithms were explained very much in detail to clear out the basics. Now that we know the flow of the pipeline and the steps for processing, has given us a brief insight on how the raw data is collected and performed to make a meaningful decision. Also, the ethics for data science/AI is something we should not forget in real

projects when we work for corporates. Before taking this class I thought that data science was more about visualization, but it has changed now. The process before the visualization is something where we will spend most of our time. In the end, this subject has a perfect name Fundamentals of Data Science.